



App Exchange REST Table Instructions

Document creation date: 09/Jun/2021

Table of contents

1 English	3
2 Spanish	7

1 English

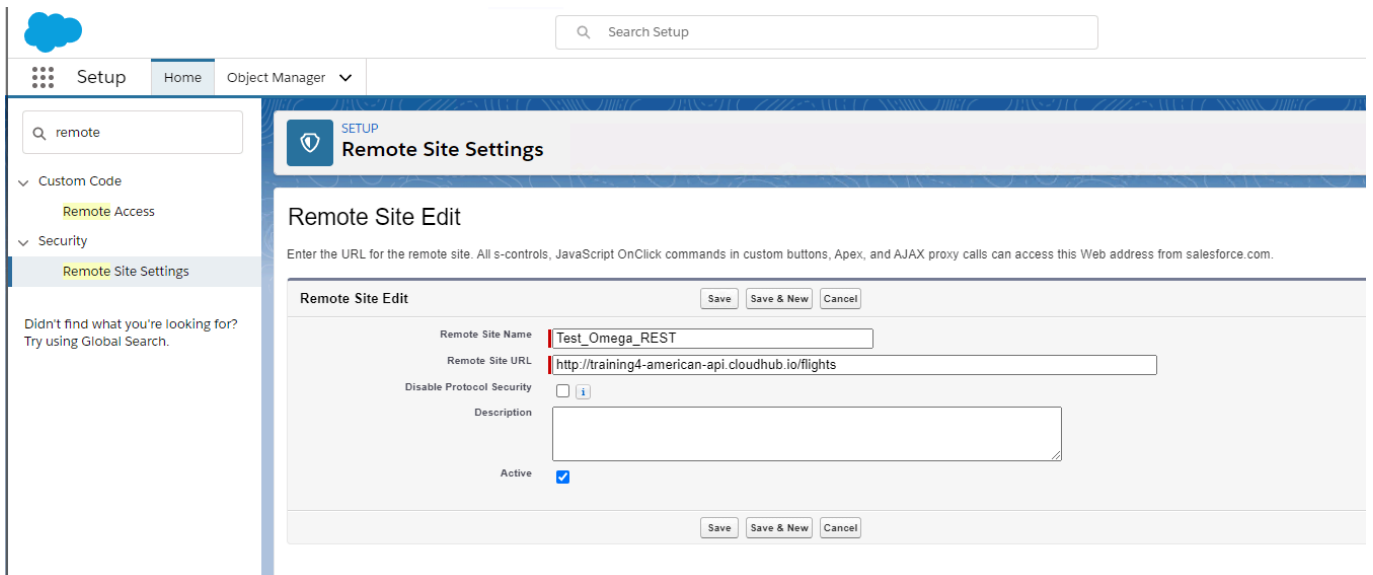
Lightning Web Component with the ability to execute a REST GET callout to obtain data from an external system. The data obtained is not saved on Salesforce, it is only available to be consulted.

Package contents:

- LWC
 - restTableContainer
 - Available for: AppPage, HomePage, RecordPage
- Labels
 - Error_DataInvalid
 - Error_NoEndpoint
 - Error_Server
 - Error_ServerEndpoint
- Apex Classes
 - RestTableController
 - RestTableController_Test
 - RestTableDesign
 - RestTableDesign_Test
- Static Resources
 - GetDataMockResponse

How to configure the component to show the data:

1- Remote Site Settings → To be able to use the endpoint, it has to be authorized. This authorization cannot be granted by the component, it has to be set from the Setup → Remote Site Settings. If you are using a Named Credential you can skip this step.

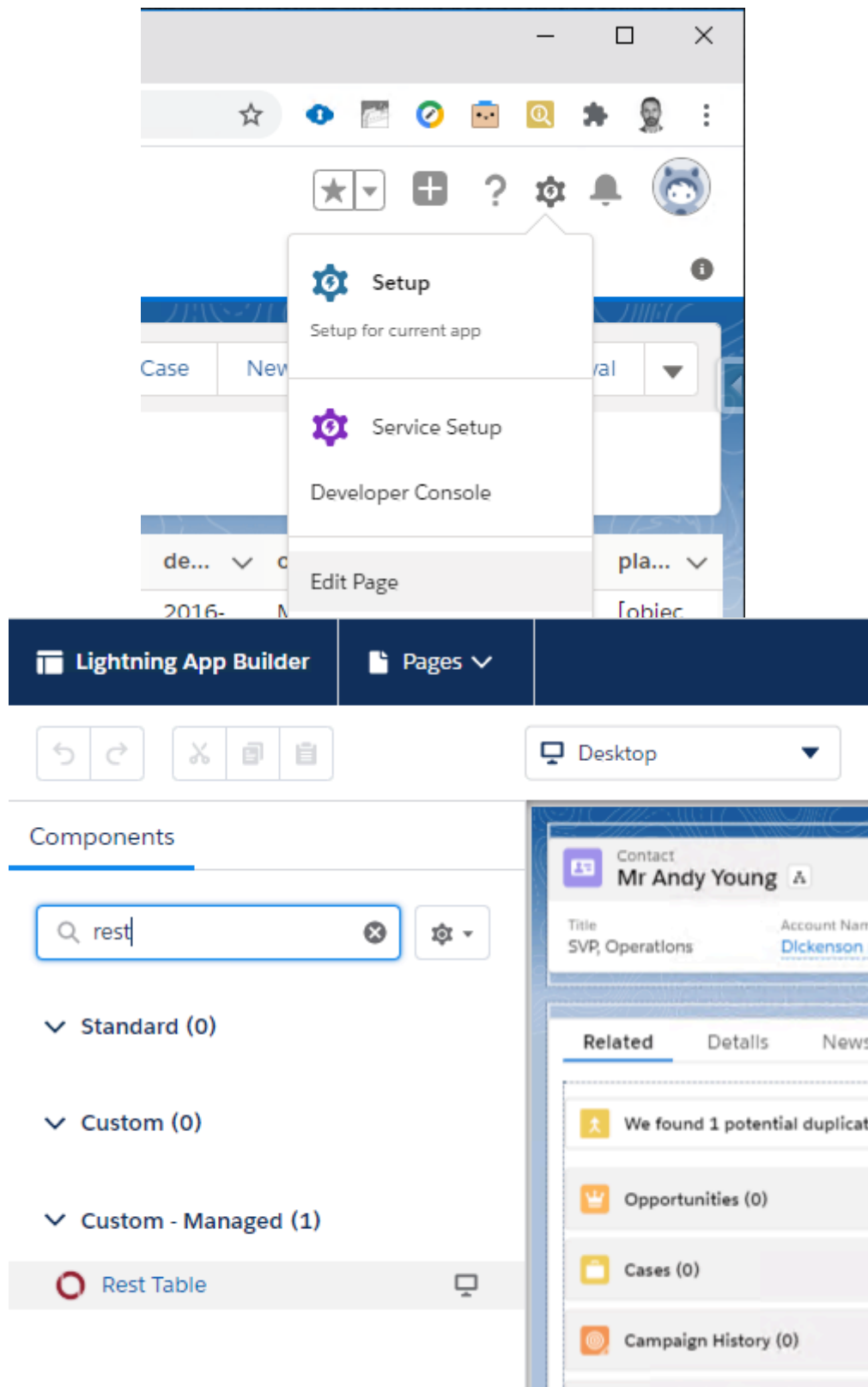


The screenshot shows the Salesforce Setup interface. The left sidebar has a search bar with 'remote' entered. Under 'Custom Code', 'Remote Access' is highlighted. Under 'Security', 'Remote Site Settings' is highlighted. The main content area is titled 'Remote Site Settings' and 'Remote Site Edit'. It contains a form with the following fields:

- Remote Site Name:** Test_Omega_REST
- Remote Site URL:** http://training4-american-api.cloudhub.io/flights
- Disable Protocol Security:** ☐ (with an information icon)
- Description:** (empty text area)
- Active:** ☒

Buttons for 'Save', 'Save & New', and 'Cancel' are present at the top and bottom of the form.

2- Add the component to the page where you want to show it (note: you must have My Domain activated)



3- Once dropped on the page where we want to show it, we must configure the connexion parameters.

Contact Record Page
← Back ? Help

Save
Activation...

+ Follow
New Case
New Note
Submit for Approval ▼

n.com

Contact Owner
 Daniel Buenestado

ID	c...	pf...	d...	or...	d...	e...	
1	rree...	541	201...	MUA	LAX	0	[obj]...
2	eefd...	300	201...	MUA	CLE	7	[obj]...
3	flee...	300	201...	MUA	LAX	0	[obj]...
4	rree...	200	201...	MUA	CLE	5	[obj]...
5	rree...	142	201...	MUA	SFO	1	[obj]...
6	flee...	954	201...	MUA	CLE	100	[obj]...
7	eefd...	676	201...	MUA	SFO	0	[obj]...
8	flee...	300	201...	MUA	SFO	30	[obj]...
9	eefd...	900	201...	MUA	SFO	0	[obj]...
10	eefd...	900	201...	MUA	LAX	100	[obj]...
11	rree...	456	201...	MUA	SFO	100	[obj]...

View Duplicates

New

New

Add to Campaign

Page > Rest Table

endpoint
 http://training4-american-api.cloudhub.io

namedCredential

recordParameterName1

recordParameterFiel1

AppPage, HomePage, RecordPage attributes:

- endpoint → REST callout address.
 - **IMPORTANT:** If you are using a named credential, only inform this field with the address appended to the Named Credential base URL
 - Example:

<https://www.myownwebservice/v2/cars/buyers>

Named Credential url: <https://www.myownwebservice>

endpoint field value: v2/cars/buyers

- `namedCredential` → The name of the Named Credential
- `headerAttributeX` → Name of the header attribute for the REST callout.
- `headerValueX` → Value for the REST callout attribute
 - Where X is a value from 1 to 5.
- `urlParameterNameX` → Name of the parameter that will go on the URL. It will concatenate it at the end of the endpoint.
- `urlParameterValueX` → Hardcoded value that will be assigned to the respective parameter on the URL.

RecordPage attributes:

- recordParameterNameX → Name of the parameter that will go on the URL. It will concatenate it at the end of the endpoint.
- recordParameterFieldX → The field from which we will extract the value that will be assign to the parameter on the URL.
 - Where X is a value from 1 to 3.

For demo purposes you can use the following:

- endpoint: <http://training4-american-api.cloudhub.io/flights>
- headerAttribute1: client_id
- headerValue1: d846d01ea59640108779eaaa532d00ca
- headerAttribute2: client_secret

- headerValue2: F7C5ecA48EBA449399b4B140416849B0

4- Once the changes are saved, the external data should be showing:

The screenshot shows a Salesforce CRM interface. At the top, there's a navigation bar with 'Accounts', 'Contacts', 'Cases', 'Reports', and 'Dashboards'. Below this, there's a search bar and a 'Search Salesforce' button. The main content area displays a contact record for 'Andrew Young'. The contact details include a phone number '(785) 241-6200', an email 'a_young@dickenson.com', and an address '1301 Hoch Drive'. To the right of the contact details, there's a table with 11 rows and 10 columns. The columns are labeled: ID, code, pri..., de..., ori..., de..., e..., and pla... The table contains data for various records, including their IDs, codes, priorities, dates, origins, destinations, and other details.

The REST GET service must return a JSON with the following list structure: `[{}, {}, {}, ..., {}]`. Where each element of the list is a JSON object with key value pairs (The keys must be the same for all the list elements). The component will read the keys and use them to give name to the datatable columns. The values will populate the datatable rows.

- Example:

```
[
  {
    "firstName": "Joe",
    "lastName": "Doe",
    "Age": 20
  },
  {
    "firstName": "Jane",
    "lastName": "Doe",
    "Age": 22
  }
]
```

Will become:

firstName	lastName	Age
Joe	Doe	20

firstName	lastName	Age
Jane	Doe	22

2 Spanish

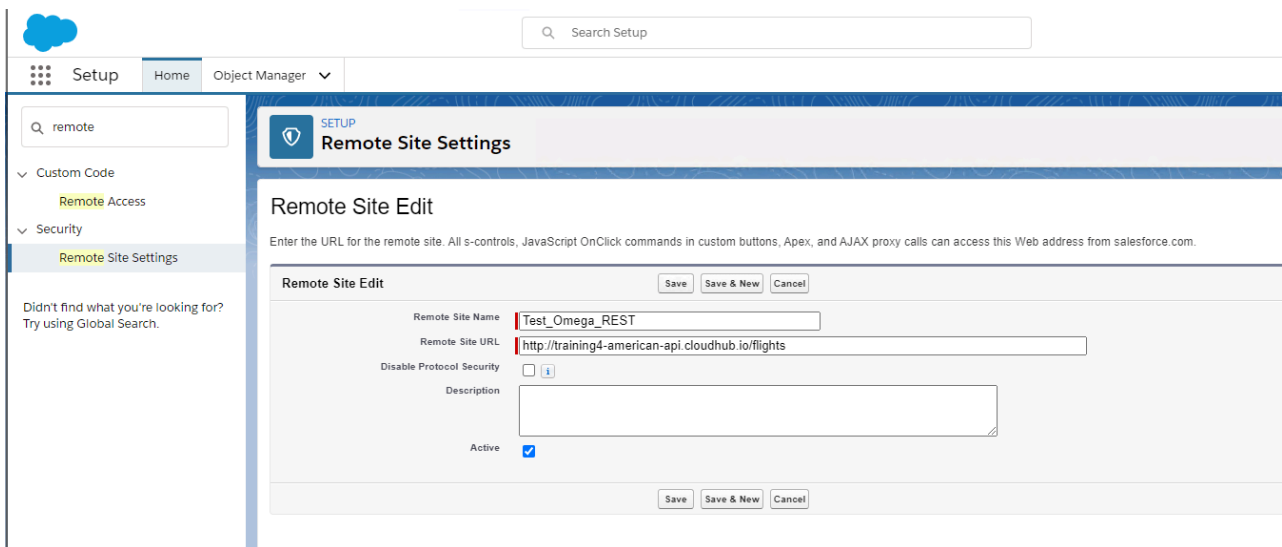
Lightning Web Component con llamada REST GET para obtener datos de un sistema externo. Los datos obtenidos no se almacenaran en Salesforce, únicamente estará disponibles para ser visualizados.

Contenido del paquete:

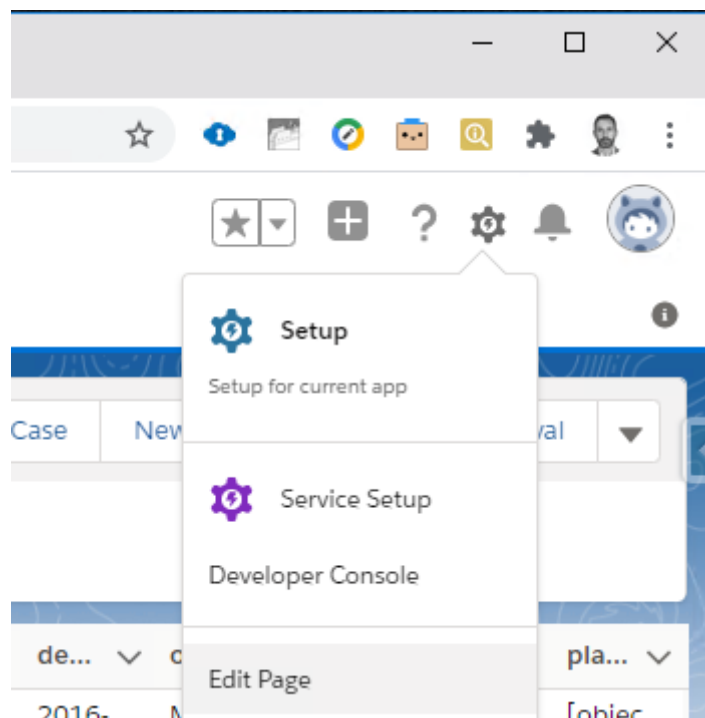
- LWC
 - restTableContainer
 - Disponible para: AppPage, HomePage, RecordPage
- Labels
 - Error_DataInvalid
 - Error_NoEndpoint
 - Error_Server
 - Error_ServerEndpoint
- Apex Classes
 - RestTableController
 - RestTableController_Test
 - RestTableDesign
 - RestTableDesign_Test
- Static Resources
 - GetDataMockResponse

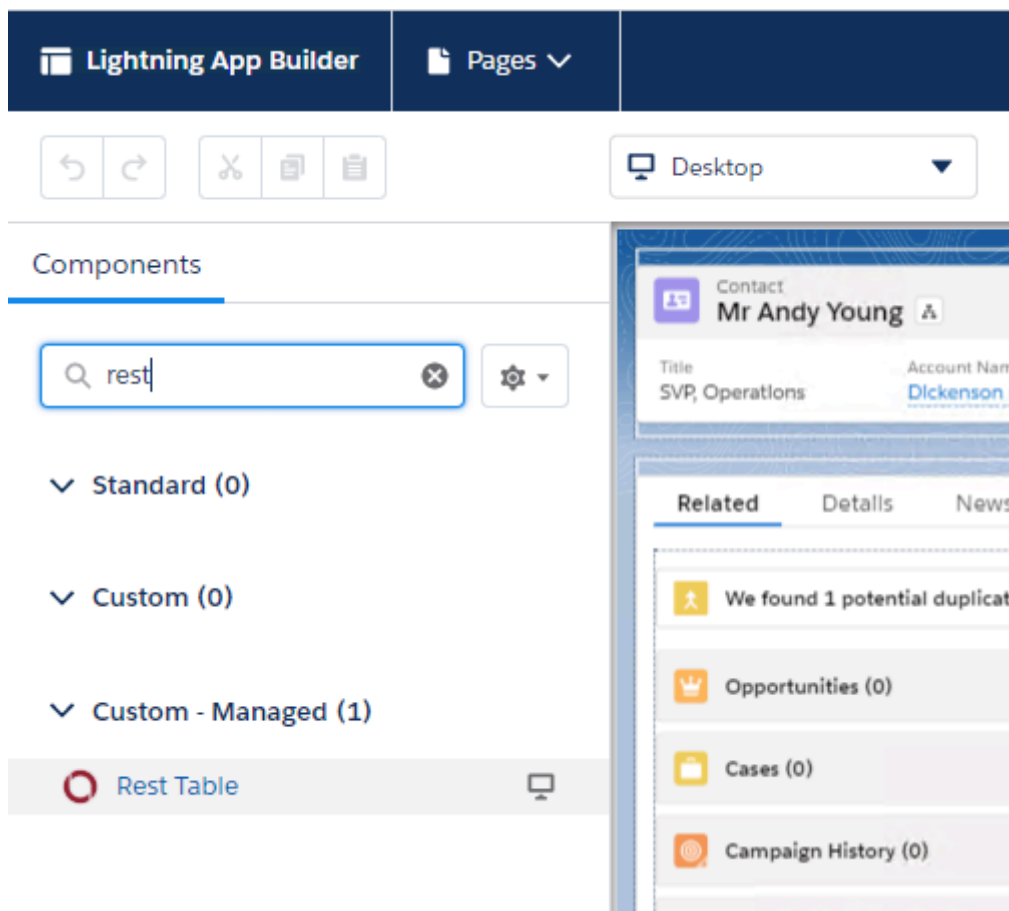
Configurar el componente para mostrar los datos:

1- Remote Site Settings → El endpoint que vaya a usarse para acceder a los datos debe estar autorizado. Esta autorización no puede gestionarse desde el componente, ha de hacerse desde Setup → Remote Site Settings. Pudes saltarte este paso si usas un Named Credential.

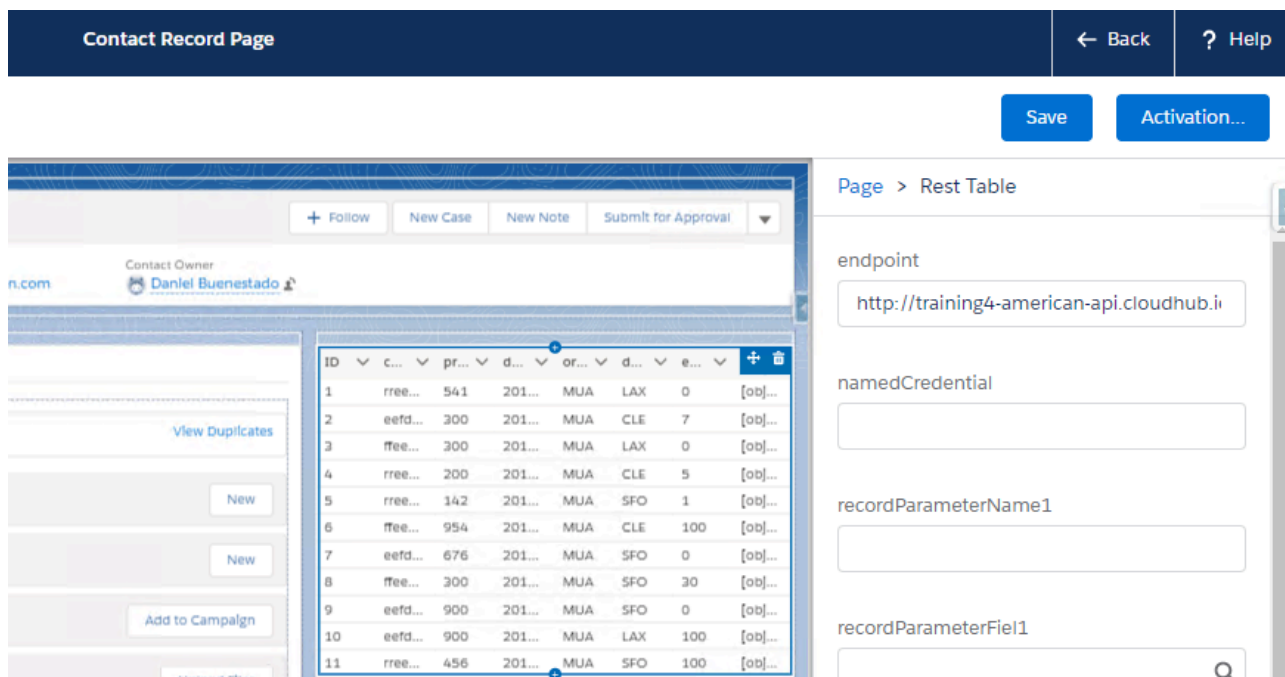


2-Añadir el componente a la página donde se quiera mostrar (nota: es necesario tener activado “My Domain” previamente)





3-Una vez arrastrado el componente a la ubicación donde lo queremos mostrar en la página, es necesario configurar los parámetros de conexión.



Tendremos los siguientes campos disponibles para configurar la llamada REST GET:

Atributos para AppPage, HomePage, RecordPage

- endpoint → Dirección a la que se hará la llamada REST.
 - IMPORTANTE: Si estás usando un Named Credential, informa únicamente de la dirección concatenada a la url de base definida en el Named Credential:
 - Ejemplo:

<https://www.myownwebservice/v2/cars/buyers>

Named Credential url: <https://www.myownwebservice>

Valor del campo endpoint: v2/cars/buyers

- headerAttributeX → Nombre del atributo para la cabecera de la llamada REST.
- headerValueX → Valor que contendrá el atributo de la cabecera.
 - Donde X es un valor de 1 a 5.
- urlParameterNameX → Nombre del parámetro que irá en la url de la llamada. Se concatenará al final del endpoint.
- urlParameterValueX → Valor fijo que se asignará al parámetro correspondiente.

Atributos para RecordPage

- recordParameterNameX → Nombre del parámetro que irá en la url de la llamada. Se concatenará al final del endpoint.
- recordParameterFieldX → Valor del parámetro correspondiente a extraer de un campo determinado del registro.
 - Donde X es un valor de 1 a 3

Para demos se pueden utilizar estos parámetros:

- endpoint: <http://training4-american-api.cloudhub.io/flights>
- headerAttribute1: client_id
- headerValue1: d846d01ea59640108779eaaa532d00ca
- headerAttribute2: client_secret
- headerValue2: F7C5ecA48EBA449399b4B140416849B0

4-Una vez guardados los cambios y activado el componente, se mostrará el componente Rest con los datos provenientes del sistema externo:

ID	code	pri...	de...	ori...	de...	e...	pla...
1	rree0...	541	2016-...	MUA	LAX	0	[obje...
2	eefd0...	300	2016-...	MUA	CLE	7	[obje...
3	ffee0...	300	2016-...	MUA	LAX	0	[obje...
4	rree1...	200	2016-...	MUA	CLE	5	[obje...
5	rree1...	142	2016-...	MUA	SFO	1	[obje...
6	ffee1...	954	2016-...	MUA	CLE	100	[obje...
7	eefd1...	676	2016-...	MUA	SFO	0	[obje...
8	ffee2...	300	2016-...	MUA	SFO	30	[obje...
9	eefd3...	900	2016-...	MUA	SFO	0	[obje...
10	eefd4...	900	2016-...	MUA	LAX	100	[obje...
11	rree4...	456	2016-...	MUA	SFO	100	[obje...

Nota técnica: el servicio REST GET debe retornar un JSON con la siguiente estructura de lista: [{},{},...,{ }]. Donde cada elemento de la lista es un objeto JSON con con parejas clave valor (las claves deberán ser iguales para todos los elementos de la lista). El componente usará las claves para dar nombre a las columnas y los valores para crear líneas de la tabla.

- Ejemplo:

```
[
  {
    "firstName": "Joe",
    "lastName": "Doe",
    "Age": 20
  },
  {
    "firstName": "Jane",
    "lastName": "Doe",
    "Age": 22
  }
]
```

Pasará a ser:

firstName	lastName	Age
Joe	Doe	20

firstName	lastName	Age
Jane	Doe	22